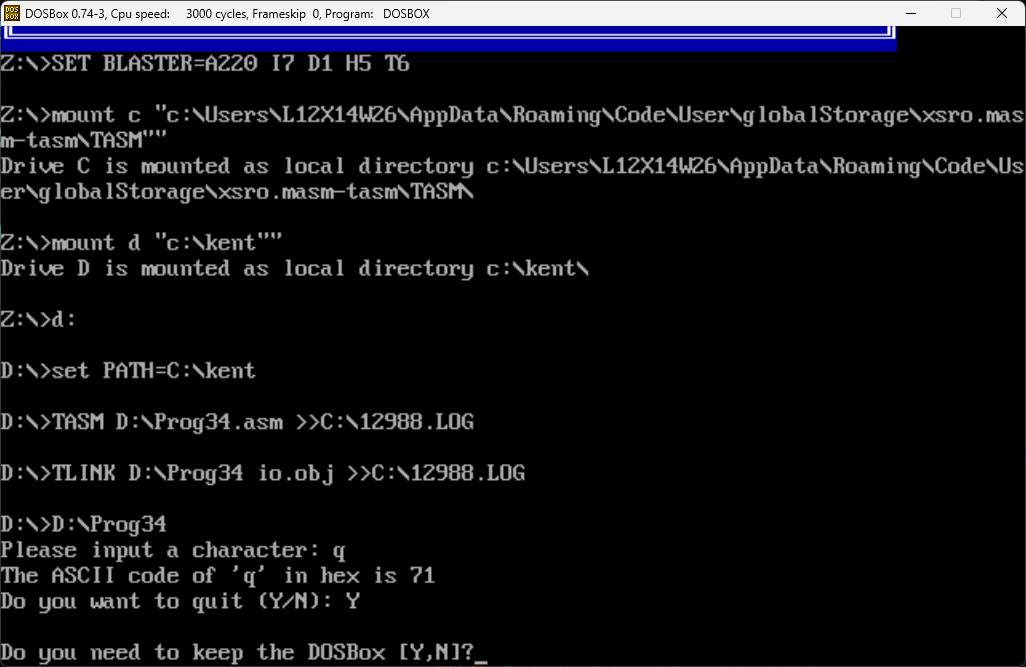
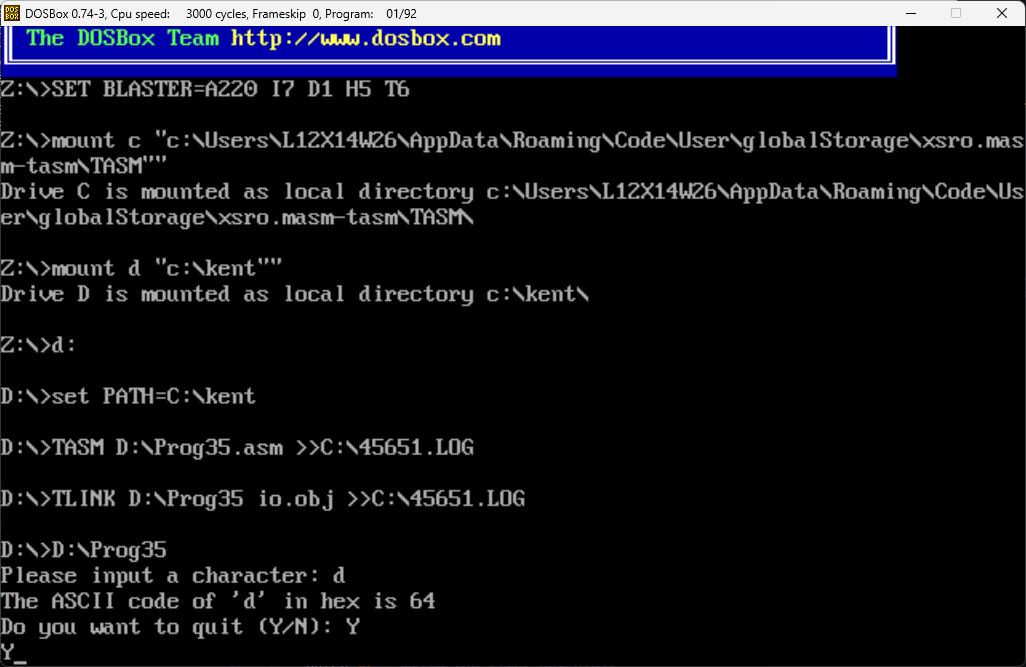
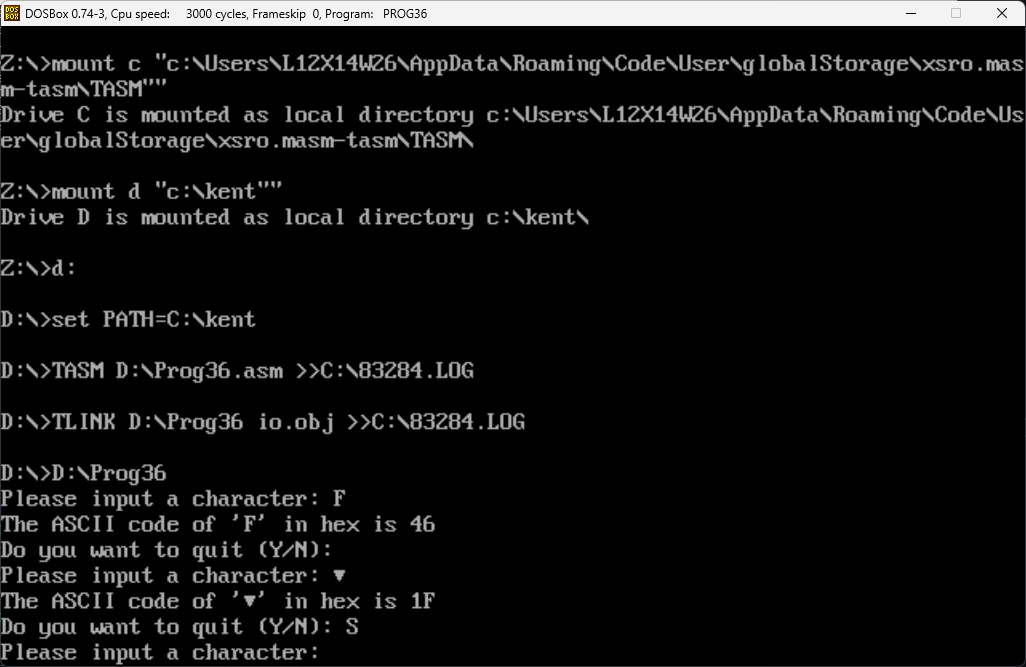
* Prog34.asm
  + Screenshot



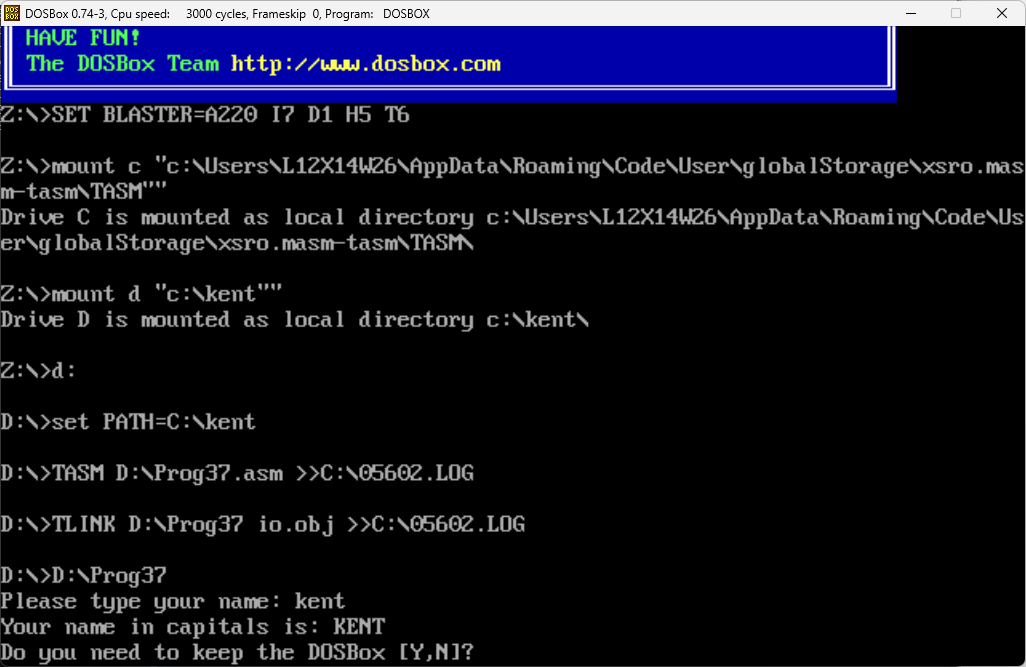
* + Code
* *; Filename: PROG34.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Description:*
* .MODEL SMALL
* .STACK 100H
* .DATA
* char\_prompt DB 'Please input a character: ',0
* out\_msg1 DB 'The ASCII code of ''',0
* out\_msg2 DB ''' in hex is ',0
* query\_msg DB 'Do you want to quit (Y/N): ',0
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* read\_char:
* PutStr char\_prompt *; request a char. input*
* GetCh AL *; read input character*
* nwln
* PutStr out\_msg1
* PutCh AL
* PutStr out\_msg2
* mov AH,AL *; save input character in AH*
* shr AL,4 *; move upper 4 bits to lower half*
* mov CX,2 *; loop count - 2 hex digits to print*
* print\_digit:
* cmp AL,9 *; if greater than 9*
* jg A\_to\_F *; convert to A through F digits*
* add AL,'0' *; otherwise, convert to 0 through 9*
* jmp skip
* A\_to\_F:
* add AL,'A'-10   *; subtract 10 and add 'A'*
* *; to convert to A through F*
* skip:
* PutCh AL *; write the first hex digit*
* mov AL,AH *; restore input character in AL*
* and AL,0FH *; mask off the upper half byte*
* loop print\_digit
* nwln
* PutStr query\_msg *; query user whether to terminate*
* GetCh AL *; read response*
* nwln
* cmp AL,'Y' *; if response is not 'Y'*
* jne read\_char *; read another character*
* done: *; otherwise, terminate program*
* mov ax, 4C00h
* int 21h
* main ENDP
* END main
* Prog35.asm
  + Screenshot



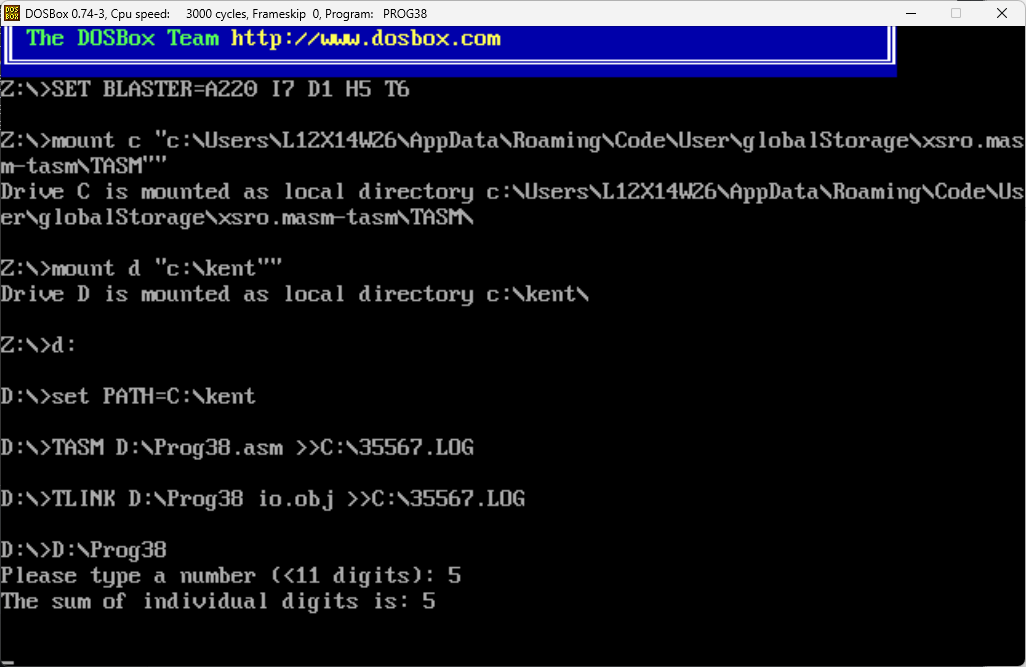
* + Code
* *; Filename: PROG35.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Description:*
* .MODEL SMALL
* .STACK 100H
* .DATA
* char\_prompt DB 'Please input a character: ',0
* out\_msg1 DB 'The ASCII code of ''',0
* out\_msg2 DB ''' in hex is ',0
* query\_msg DB 'Do you want to quit (Y/N): ',0
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* read\_char:
* PutStr char\_prompt *; request a char. input*
* GetCh AL *; read input character*
* nwln
* PutStr out\_msg1
* PutCh AL
* PutStr out\_msg2
* mov AH,AL *; save input character in AH*
* shr AL,4 *; move upper 4 bits to lower half*
* mov CX,2 *; loop count - 2 hex digits to print*
* print\_digit:
* cmp AL,9 *; if greater than 9*
* jg A\_to\_F *; convert to A through F digits*
* add AL,'0' *; otherwise, convert to 0 through 9*
* jmp skip
* A\_to\_F:
* add AL,'A'-10 *; subtract 10 and add 'A'*
* *; to convert to A through F*
* skip:
* PutCh AL *; write the first hex digit*
* mov AL,AH *; restore input character in AL*
* and AL,0FH *; mask off the upper half byte*
* loop print\_digit
* nwln
* PutStr query\_msg *; query user whether to terminate*
* GetCh AL *; read response*
* nwln
* cmp AL,'Y' *; if response is not 'Y'*
* jne read\_char *; read another character*
* done: *; otherwise, terminate program*
* main ENDP
* END main
* Prog36.asm
  + Screenshot



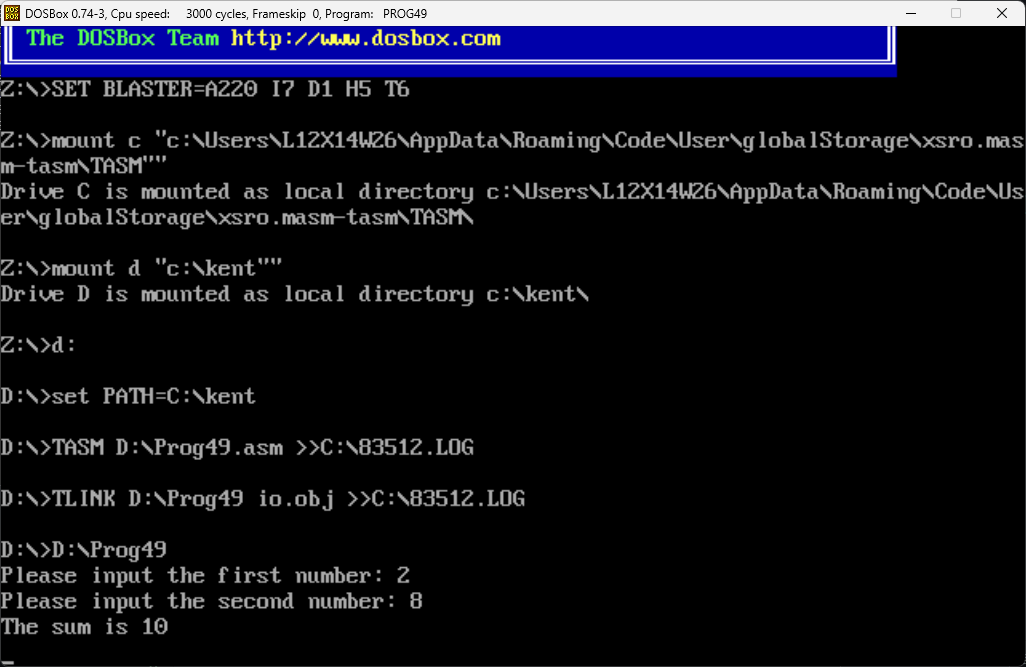
* + Code
* *; Filename: PROG36.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Description:*
* .MODEL SMALL
* .STACK 100H
* .DATA
* char\_prompt    DB 'Please input a character: ',0
* out\_msg1       DB 'The ASCII code of ''',0
* out\_msg2       DB ''' in hex is ',0
* query\_msg      DB 'Do you want to quit (Y/N): ',0
* *; translation table: 4-bit binary to hex*
* hex\_table      DB '0123456789ABCDEF'
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* read\_char:
* PutStr char\_prompt        *; request a char. input*
* GetCh AL                  *; read input character*
* nwln
* PutStr out\_msg1
* PutCh AL
* PutStr out\_msg2
* mov AH,AL                 *; save input character in AH*
* mov BX,OFFSET hex\_table   *; BX := translation table*
* shr AL,4                  *; move upper 4 bits to lower half*
* xlatb                      *; replace AL with hex digit*
* PutCh AL                  *; write the first hex digit*
* mov AL,AH                 *; restore input character to AL*
* and AL,0FH                *; mask off upper 4 bits*
* xlatb
* PutCh AL                  *; write the second hex digit*
* nwln
* PutStr query\_msg          *; query user whether to terminate*
* GetCh AL                  *; read response*
* nwln
* cmp AL,'Y'                *; if response is not 'Y'*
* jne read\_char             *; read another character*
* done:                          *; otherwise, terminate program*
* main ENDP
* END main
* Prog37.asm
  + Screenshot



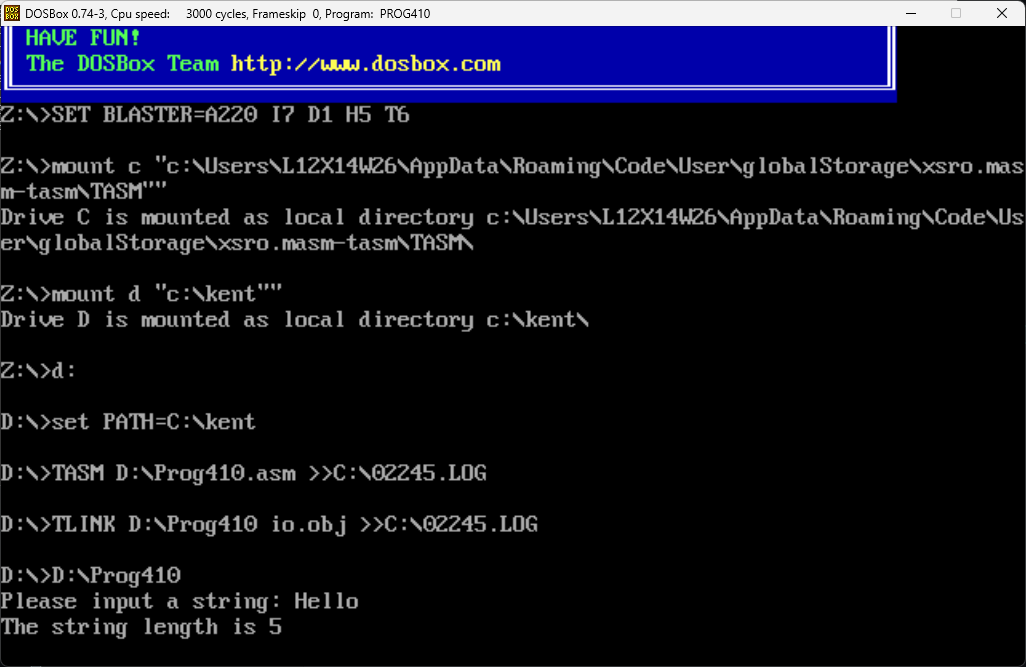
* + Code
* *; Filename: PROG37.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Description:*
* .MODEL SMALL
* .STACK 100H
* .DATA
* name\_prompt    DB 'Please type your name: ',0
* out\_msg        DB 'Your name in capitals is: ',0
* in\_name        DB 31 DUP (?)
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr name\_prompt        *; request character string*
* GetStr in\_name,31         *; read input character string*
* nwln
* PutStr out\_msg
* mov BX,OFFSET in\_name     *; BX := address of in\_name*
* process\_char:
* mov AL,[BX]               *; move the char. to AL*
* cmp AL,0                  *; if it is the NULL character*
* je done                   *; conversion done*
* cmp AL,'a'                *; if (char < 'a')*
* jl not\_lower\_case         *; not a lowercase letter*
* cmp AL,'z'                *; if (char > 'z')*
* jg not\_lower\_case         *; not a lowercase letter*
* lower\_case:
* add AL,'A'-'a'            *; convert to uppercase*
* not\_lower\_case:
* PutCh AL                  *; write the character*
* inc BX                    *; BX points to next char.*
* jmp process\_char          *; go back to process next char.*
* nwln
* done:
* mov ax, 4C00h
* int 21h
* main ENDP
* END main
* Prog38.asm
  + Screenshot



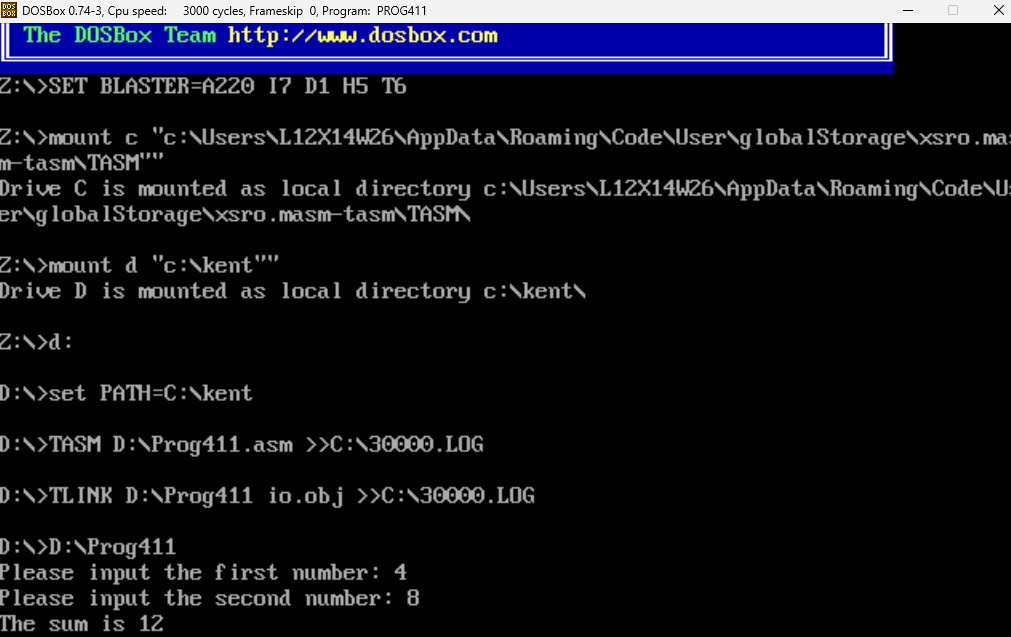
* + Code
* *; Filename: PROG38.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Description:*
* .MODEL SMALL
* .STACK 100H
* .DATA
* number\_prompt    DB 'Please type a number (<11 digits): ',0
* out\_msg          DB 'The sum of individual digits is: ',0
* number           DB 11 DUP (?)
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr number\_prompt     *; request an input number*
* GetStr number,11         *; read input number as a string*
* nwln
* mov BX,OFFSET number     *; BX := address of number*
* sub DX,DX                *; DX := 0 -- DL keeps the sum*
* repeat\_add:
* mov AL,[BX]              *; move the digit to AL*
* cmp AL,0                 *; if it is the NULL character*
* je done                  *; sum is done*
* and AL,0FH               *; mask off the upper 4 bits*
* add DL,AL                *; add the digit to sum*
* inc BX                   *; increment BX to point to next digit*
* jmp repeat\_add           *; and jump back*
* done:
* PutStr out\_msg
* PutInt DX                *; write sum*
* nwln
* main ENDP
* END main
* Prog49.asm
  + Screenshot



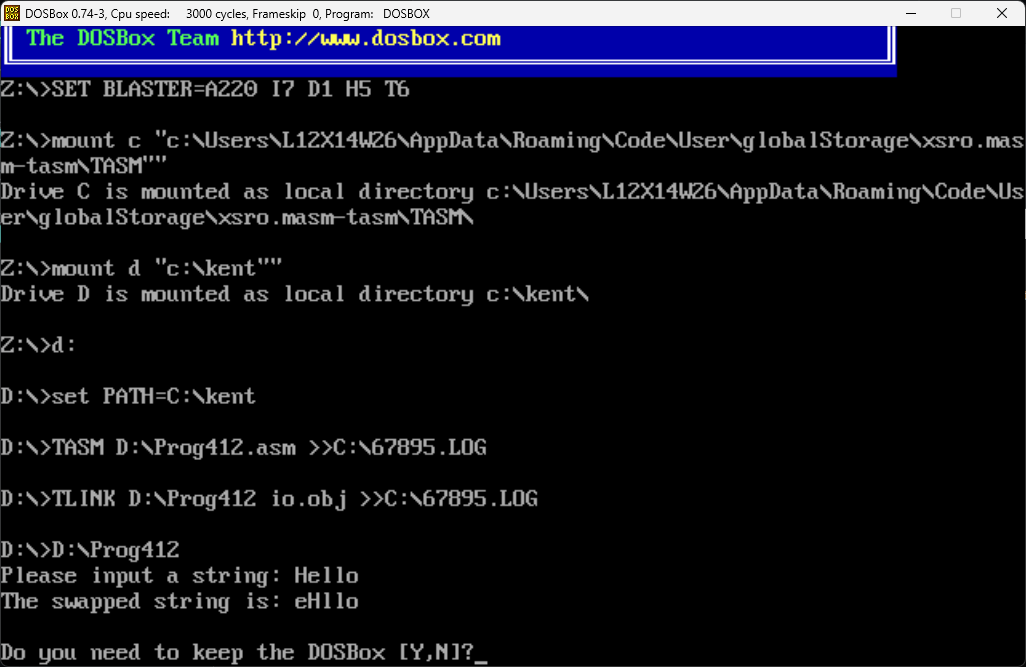
* + Code
* *; Filename: PROG49.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Objective: To show parameter passing via registers*
* *; Input: Requests two integers from the user.*
* *; Output: Outputs the sum of the input integers.*
* .MODEL SMALL
* .STACK 100H
* .DATA
* prompt\_msg1 DB 'Please input the first number: ',0
* prompt\_msg2 DB 'Please input the second number: ',0
* sum\_msg DB 'The sum is ',0
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr prompt\_msg1 *; request first number*
* GetInt CX *; CX := first number*
* nwln
* PutStr prompt\_msg2 *; request second number*
* GetInt DX *; DX := second number*
* nwln
* call sum *; returns sum in AX*
* PutStr sum\_msg *; display sum*
* PutInt AX
* nwln
* done:
* main ENDP
* *;--------------------------------------------------*
* *;Procedure sum receives two integers in CX and DX.*
* *; The sum of the two integers is returned in AX.*
* *;--------------------------------------------------*
* sum PROC
* mov AX,CX *; sum := first number*
* add AX,DX *; sum := sum + second number*
* ret
* sum ENDP
* END main
* Prog410.asm
  + Screenshot



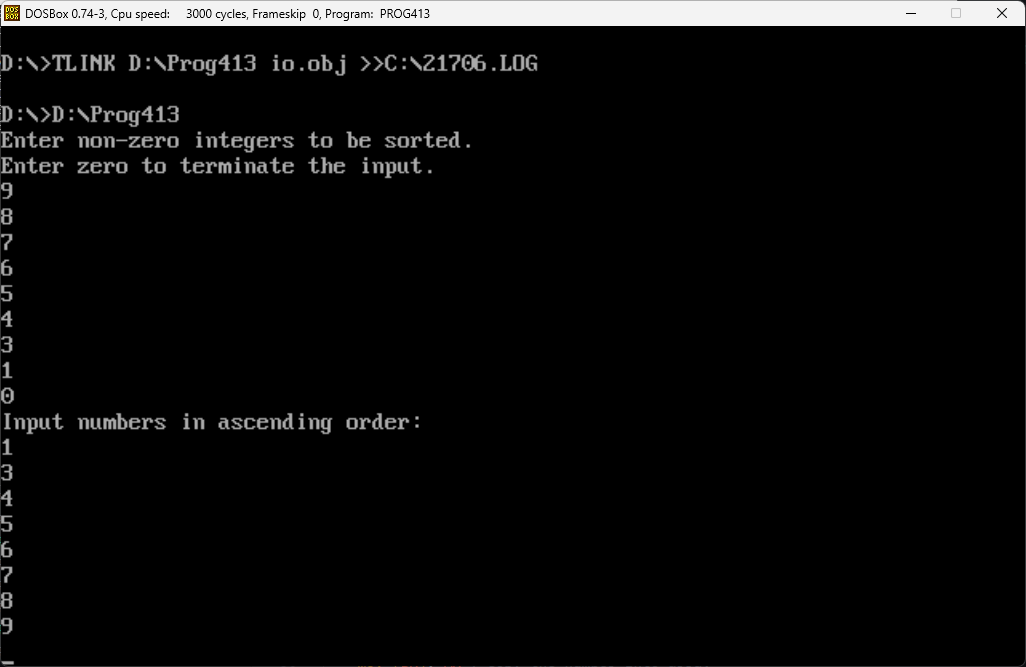
* + Code
* *; Filename: PROG410.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Objective: To show parameter passing via registers*
* *; Input: Requests a character string from the user.*
* *; Output: Outputs the length of the input string.*
* BUF\_LEN EQU 41 *; string buffer length*
* .MODEL SMALL
* .STACK 100H
* .DATA
* string DB BUF\_LEN DUP (?) *; input string < BUF\_LEN chars.*
* prompt\_msg DB 'Please input a string: ', 0
* length\_msg DB 'The string length is ', 0
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr prompt\_msg *; request string input*
* GetStr string, BUF\_LEN *; read string from keyboard*
* nwln
* mov BX, OFFSET string *; BX := string address*
* call str\_len *; returns string length in AX*
* PutStr length\_msg *; display string length*
* PutInt AX
* nwln
* done:
* main ENDP
* *;-----------------------------------------------------------*
* *; Procedure str\_len receives a pointer to a string in BX.*
* *; String length is returned in AX.*
* *;-----------------------------------------------------------*
* str\_len PROC
* push BX
* sub AX, AX *; string length := 0*
* repeat:
* cmp BYTE PTR [BX], 0 *; compare with NULL char.*
* je str\_len\_done *; if NULL we are done*
* inc AX *; else, increment string length*
* inc BX *; point BX to the next char.*
* jmp repeat *; and repeat the process*
* str\_len\_done:
* pop BX
* ret
* str\_len ENDP
* END main
* Prog411.asm
  + Screenshot



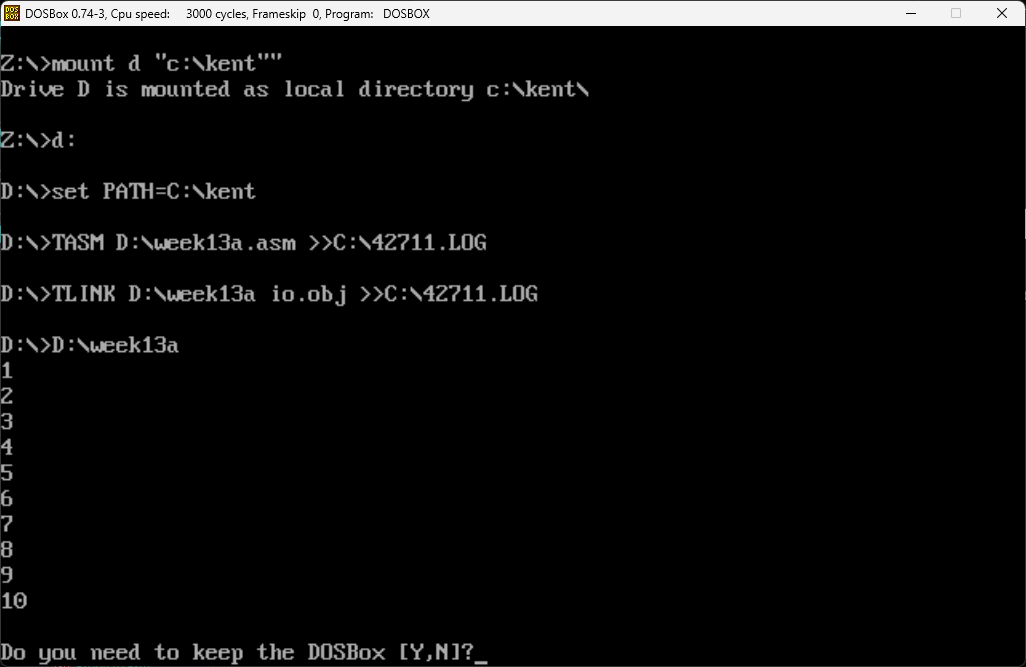
* + Code
* *; Filename: PROG411.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Objective: To show parameter passing via the stack*
* *; Input: Requests two integers from the user.*
* *; Output: Outputs the sum of the input integers.*
* .MODEL SMALL
* .STACK 100H
* .DATA
* prompt\_msg1 DB 'Please input the first number: ', 0
* prompt\_msg2 DB 'Please input the second number: ', 0
* sum\_msg DB 'The sum is ', 0
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr prompt\_msg1 *; request first number*
* GetInt CX *; CX := first number*
* nwln
* PutStr prompt\_msg2 *; request second number*
* GetInt DX *; DX := second number*
* nwln
* push CX *; place first number on stack*
* push DX *; place second number on stack*
* call sum *; returns sum in AX*
* PutStr sum\_msg *; display sum*
* PutInt AX
* nwln
* done:
* main ENDP
* *;-----------------------------------------------------------*
* *; Procedure sum receives two integers via the stack.*
* *; The sum of the two integers is returned in AX.*
* *;-----------------------------------------------------------*
* sum PROC
* push BP *; we will use BP, so save it*
* mov BP, SP
* mov AX, [BP+6] *; sum := first number*
* add AX, [BP+4] *; sum := sum + second number*
* pop BP *; restore BP*
* ret 4 *; return and clear parameters*
* sum ENDP
* END main
* Prog412.asm
  + Screenshot



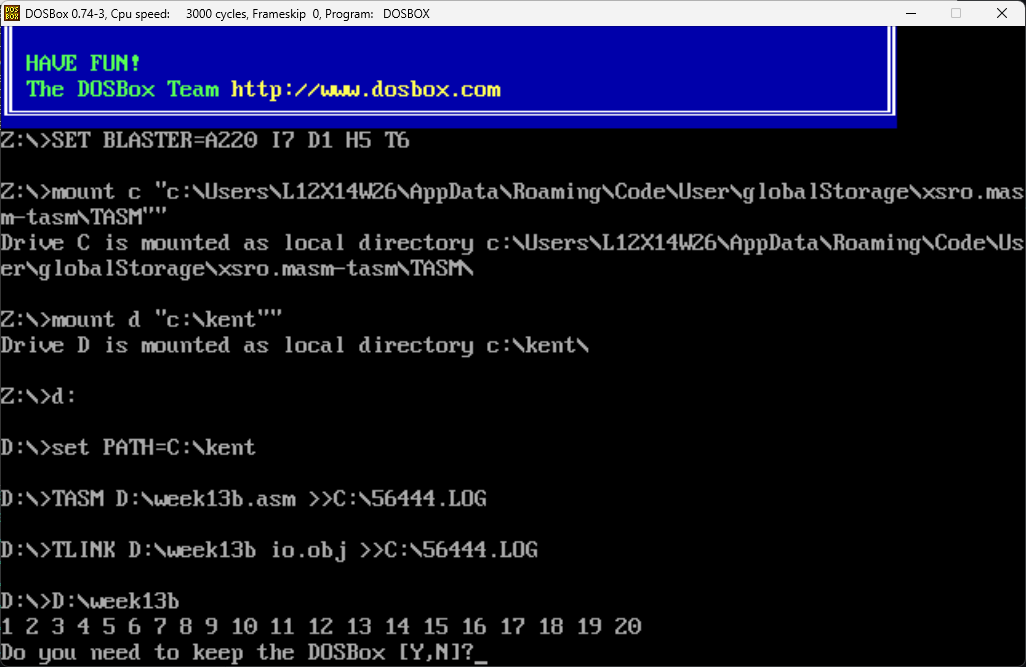
* + Code
* *; Filename: PROG412.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Objective: To show parameter passing via the stack*
* *; Input: Requests a character string from the user.*
* *; Output: Outputs the input string with the first two characters swapped.*
* BUF\_LEN EQU 41 *; string buffer length*
* .MODEL SMALL
* .STACK 100H
* .DATA
* string DB BUF\_LEN DUP (?) *; input string < BUF\_LEN chars.*
* prompt\_msg DB 'Please input a string: ', 0
* output\_msg DB 'The swapped string is: ', 0
* .CODE
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr prompt\_msg *; request string input*
* GetStr string, BUF\_LEN *; read string from the user*
* nwln
* mov AX, OFFSET string *; AX := string[0] pointer*
* push AX *; push string[0] pointer on stack*
* inc AX *; AX := string[1] pointer*
* push AX *; push string[1] pointer on stack*
* call swap *; swaps the first two characters*
* PutStr output\_msg *; display the swapped string*
* PutStr string
* nwln
* done:
* mov AX, 4C00h
* int 21h
* main ENDP
* *;-----------------------------------------------------------*
* *; Procedure swap receives two pointers (via the stack) to*
* *; characters of a string. It exchanges these two characters.*
* *;-----------------------------------------------------------*
* swap PROC
* push BP *; save BP - procedure uses BP*
* mov BP, SP *; copy SP to BP*
* push BX *; save BX - procedure uses BX*
* *; swap begins here. Because of xchg, AL is preserved.*
* mov BX, [BP+6] *; BX := first character pointer*
* xchg AL, [BX]
* mov BX, [BP+4] *; BX := second character pointer*
* xchg AL, [BX]
* mov BX, [BP+6] *; BX := first character pointer*
* xchg AL, [BX]
* *; swap ends here*
* pop BX *; restore registers*
* pop BP
* ret 4 *; return and clear parameters*
* swap ENDP
* END main
* Prog413.asm
  + Screenshot



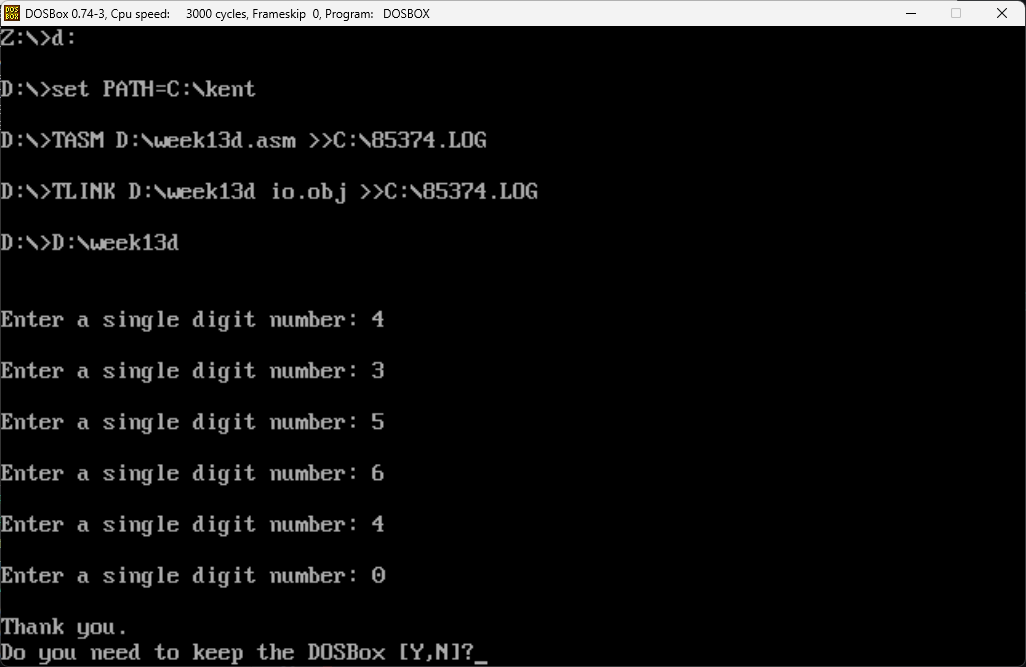
* + Code
* *; Filename: PROG13.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* *; Objective: To implement the bubble sort algorithm*
* *; Input: A set of non-zero integers to be sorted.*
* *; Input is terminated by entering zero.*
* *; Output: Outputs the numbers in ascending order.*
* CRLF EQU 0DH, 0AH
* MAX\_SIZE EQU 20
* .MODEL SMALL
* .STACK 100H
* .DATA
* array DW MAX\_SIZE DUP (?) *; input array for integers*
* prompt\_msg DB 'Enter non-zero integers to be sorted.', CRLF
* DB 'Enter zero to terminate the input.', 0
* output\_msg DB 'Input numbers in ascending order:', 0
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* PutStr prompt\_msg *; request input numbers*
* nwln
* mov BX, OFFSET array *; BX := array pointer*
* mov CX, MAX\_SIZE *; CX := array size*
* sub DX, DX *; number count := 0*
* read\_loop:
* GetInt AX *; read input number*
* nwln
* cmp AX, 0 *; if the number is zero*
* je stop\_reading *; no more numbers to read*
* mov [BX], AX *; copy the number into array*
* add BX, 2 *; BX points to the next element*
* inc DX *; increment number count*
* loop read\_loop *; reads a max. of MAX\_SIZE numbers*
* stop\_reading:
* push DX *; push array size onto stack*
* push OFFSET array *; place array pointer on stack*
* call bubble\_sort
* PutStr output\_msg *; display sorted input numbers*
* nwln
* mov BX, OFFSET array
* mov CX, DX *; CX := number count*
* print\_loop:
* PutInt [BX]
* nwln
* add BX, 2
* loop print\_loop
* done:
* main ENDP
* *;-----------------------------------------------------------*
* *; This procedure receives a pointer to an array of integers*
* *; and the size of the array via the stack. It sorts the*
* *; array in ascending order using the bubble sort algorithm.*
* *;-----------------------------------------------------------*
* SORTED EQU 0
* UNSORTED EQU 1
* bubble\_sort PROC
* pusha
* mov BP, SP
* *; CX serves the same purpose as the end\_index variable*
* *; in the C procedure. CX keeps the number of comparisons*
* *; to be done in each pass. Note that CX is decremented*
* *; by 1 after each pass.*
* mov CX, [BP + 20] *; load array size into CX*
* mov BX, [BP + 18] *; load array address into BX*
* next\_pass:
* dec CX *; if # of comparisons is zero*
* jz sort\_done *; then we are done*
* mov DI, CX *; else start another pass*
* *; DX is used to keep SORTED/UNSORTED status*
* mov DX, SORTED *; set status to SORTED*
* *; SI points to element X and SI+2 to the next element*
* mov SI, BX *; load array address into SI*
* pass:
* *; This loop represents one pass of the algorithm.*
* *; Each iteration compares elements at [SI] and [SI+2]*
* *; and swaps them if ([SI]) < ([SI+2]).*
* mov AX, [SI]
* cmp AX, [SI + 2]
* jg swap
* increment:
* *; Increment SI by 2 to point to the next element*
* add SI, 2
* dec DI
* jnz pass
* cmp DX, SORTED *; if status remains SORTED*
* je sort\_done *; then sorting is done*
* jmp next\_pass *; else initiate another pass*
* swap:
* *; swap elements at [SI] and [SI+2]*
* xchg AX, [SI + 2]
* mov [SI], AX
* mov DX, UNSORTED *; set status to UNSORTED*
* jmp increment
* sort\_done:
* popa
* ret 4 *; return and clear parameters*
* bubble\_sort ENDP
* END main
* Week13a.asm
  + Screenshot



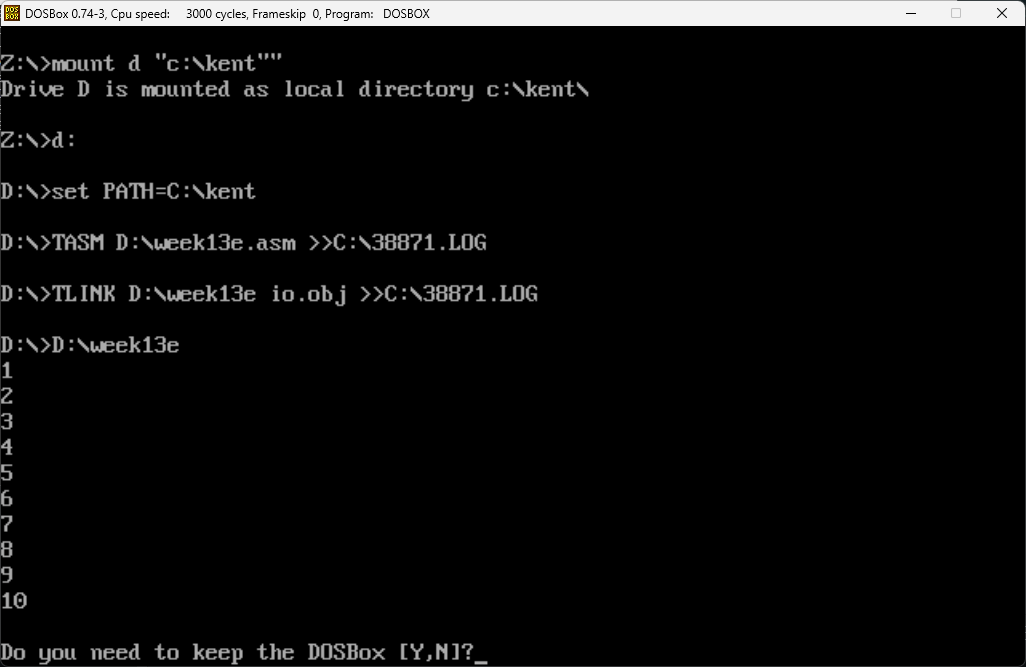
* + Code
* *; Filename: WEEK13A.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* MAX\_SIZE EQU 10
* .MODEL SMALL
* .STACK 100H
* .DATA
* newline db 0Ah      *; Newline character (ASCII 10)*
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* mov CX, 1           *; Start the counter at 1*
* print\_loop:
* cmp CX, MAX\_SIZE+1  *; Compare the counter with 11 (MAX\_SIZE + 1)*
* jge done             *; If CX >= 11, we're done*
* *; Print the current value of CX (integer)*
* PutInt CX            *; Use the PutInt macro to print the number*
* nwln                 *; Print newline after each number*
* inc CX               *; Increment CX to print the next number*
* jmp print\_loop       *; Repeat the loop*
* done:
* *; Exit the program*
* mov AX, 4C00h        *; DOS interrupt to terminate the program*
* int 21h
* main ENDP
* END main
* Week13b.asm
  + Screenshot



* + Code
* *; Filename: WEEK13B.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* MAX\_SIZE EQU 20
* .MODEL SMALL
* .STACK 100H
* .DATA
* space db ' '         *; Space character (ASCII 32)*
* .CODE
* .486
* INCLUDE io.mac
* main PROC
* .STARTUP
* mov CX, 1           *; Start the counter at 1*
* print\_loop:
* cmp CX, MAX\_SIZE+1  *; Compare the counter with 11 (MAX\_SIZE + 1)*
* jge done             *; If CX >= 11, we're done*
* *; Print the current value of CX (integer)*
* PutInt CX            *; Use the PutInt macro to print the number*
* mov ah, 02h
* mov dl, space
* int 21h
* inc CX               *; Increment CX to print the next number*
* jmp print\_loop       *; Repeat the loop*
* done:
* *; Exit the program*
* mov AX, 4C00h        *; DOS interrupt to terminate the program*
* int 21h
* main ENDP
* END main
* Week13c.asm
  + Screenshot
  + Code
* Week13d.asm
  + Screenshot



* + Code
* *; Filename: WEEK13D.ASM*
* *; Programmer Name: KENT A. RATO*
* *; Date: November 8, 2024*
* .model small
* .stack 100h
* .data
* prompt db "Enter a single digit number: $"
* thankYou db "Thank you.$"
* nl db 0Ah, 0Dh, "$"  *; Newline for output*
* .code
* main:
* *; Initialize data segment*
* mov ax, @data
* mov ds, ax
* input\_loop:
* lea dx, nl
* mov ah, 09h
* int 21h
* lea dx, nl
* mov ah, 09h
* int 21h
* *; Display the prompt*
* mov ah, 09h
* lea dx, prompt
* int 21h
* *; Read the user input (single character input)*
* mov ah, 01h
* int 21h           *; Read character into AL*
* *; Convert ASCII to integer (subtract 30h to get actual number)*
* sub al, '0'
* *; Check if the entered number is 0*
* cmp al, 0
* je thank\_you       *; If the number is 0, jump to thank\_you*
* *; If the number is not zero, loop and ask for input again*
* jmp input\_loop
* thank\_you:
* lea dx, nl
* mov ah, 09h
* int 21h
* lea dx, nl
* mov ah, 09h
* int 21h
* *; Display "Thank you."*
* mov ah, 09h
* lea dx, thankYou
* int 21h
* *; Exit the program*
* mov ah, 4Ch
* int 21h
* end main
* Week13e.asm
  + Screenshot



* + Code

*; Filename: WEEK13E.ASM*

*; Programmer Name: KENT A. RATO*

*; Date: November 8, 2024*

MAX\_SIZE EQU 10

.MODEL SMALL

.STACK 100H

.DATA

    newline db 0Ah      *; Newline character (ASCII 10)*

.CODE

.486

INCLUDE io.mac

main PROC

    .STARTUP

    mov CX, 1           *; Start the counter at 1*

print\_loop:

    cmp CX, MAX\_SIZE+1  *; Compare the counter with 11 (MAX\_SIZE + 1)*

    jge done             *; If CX >= 11, we're done*

*; Print the current value of CX (integer)*

    PutInt CX            *; Use the PutInt macro to print the number*

    nwln                 *; Print newline after each number*

    inc CX               *; Increment CX to print the next number*

    jmp print\_loop       *; Repeat the loop*

done:

*; Exit the program*

    mov AX, 4C00h        *; DOS interrupt to terminate the program*

    int 21h

main ENDP

END main